Computer Organization and Programming - Assignment #2

Matthias Bonne

Submission Deadline: May 27, 2020

1 Goals

In this assignment you will add a feature to an existing project. In the process, you will learn how to build a package from source and install it to a local directory.

2 Background

Evince is a popular lightweight document viewer for Linux. It supports several document formats, including PDF and PostScript.

One of Evince's features is its ability to "remember" certain parameters for every document that it opens. For example, when the user closes a document on page N, Evince stores the page number N in a local database. The next time the user opens the same document, Evince will open the document on page N. Thus, the user can continue reading the document from the point they left.

As an exception, if the user closes a document on the *last* page, and then opens the same document, Evince will open the document on the *first* page. This is because closing the document on the last page typically means that the user has finished reading the document. When the user opens the document again, they likely want to re-read it from the beginning.

This behavior may be useful when *reading* documents. When *writing* documents, however, users often work on the last page, and want to view it many times as they change it and add content. In this case, it can be rather annoying to have to jump to the last page every time the user wants to view their changes. We would like Evince to open such documents on the page they were closed on, even if this is the last page in the document.

In this assignment you will add a feature called "Creator Mode". When enabled, Evince will always open a document on the page it was closed on, even if this is the last page. When disabled, Evince will exhibit the old behavior as described above. The user will be able to turn this feature on and off on a per-file basis and to set its default state, both from within Evince and from the command line using external tools.

3 Preparations

In this section you will prepare the system for building a modified version of Evince. Please follow this section carefully, and run the commands in the specified order. Failure to do so may cause problems later. Note that some of the commands can take some time.

3.1 Install Dependencies

Building Evince requires some development packages. Install these dependencies by running the following commands:

```
sudo apt-get update
sudo apt-get -y build-dep evince
```

3.2 Obtain the Source Package

First, create a new directory for this project:

```
mkdir ~/evince
cd ~/evince
```

Next, download the source package. As of this writing, the latest version of Evince is 3.36.0. Search online and find the package. You should look for a file named `evince-3.36.0.tar.xz`. Download it and put it in the directory `~/evince` that you just created.

Note

Do not clone Evince's Git repository. Source code repositories usually do not contain auto-generated files that are needed for building the package. It is easier to use official source packages when they are available. You should download the source package and use it as described here.

3.3 Extract the Source Package

Extract the package:

```
tar xf evince-3.36.0.tar.xz
```

If you are familiar with a version control system, such as Git, you can use it (but see the previous note).

If not, create a copy of the original source tree:

```
mv evince-3.36.0{,-orig}
cp -a evince-3.36.0-{orig,new}
```

You now have two copies of the source code. The directory `~/evince/evince-3.36.0-orig` contains the original files. You should not modify it; it will be used when you submit your work. Your changes will be made to files under the directory `~/evince/evince-3.36.0-new`.

3.4 Build the Package

We will build Evince in a separate directory, so that the source tree remains clean. Create this directory:

```
mkdir build
cd build
```

Prepare the package for compilation:

```
../evince-3.36.0-new/configure                \
                --prefix="$HOME/evince/root" \
                --disable-dbus               \
                --disable-nautilus           \
                --without-gspell
```

The package will be installed in `~/evince/root`. The other switches disable some features that may cause errors during the build or at runtime.

Now, compile and install the package:

```
make
make install
```

Evince is now installed in `~/evince/root`.

3.5 Set Up the Environment

Since we installed Evince in a non-standard directory, we need to update some environment variables in order to use it. This is usually done in `~/.profile`.

Run the following commands:

```
echo 'PATH="$HOME/evince/root/bin:$PATH"' >> ~/.profile
echo 'XDG_DATA_DIRS="$HOME/evince/root/share:$XDG_DATA_DIRS"' >> ~/.profile
```

From now on, these environment variables will be set appropriately every time you login.

Now, tell the shell to re-read `~/.profile`, so that your changes will take effect immediately:

```
source ~/.profile
```

3.6 Run Evince

Get a PDF file and open it in Evince:

```
evince <PDF file>
```

Scroll to the second page in the document and close it. Open it again, and see that it opens on the second page as expected. Now scroll to the last page in the document and close it. Open it again, and see that it opens on the first page, as explained above.

4 Modifying the Source Code

This section specifies the changes you need to make to Evince's source code.

Note

All of your changes should be made to files under `~/evince/evince-3.36.0-new`. Do not modify any of the files under `~/evince/evince-3.36.0-orig`.

4.1 File-Specific Attributes

Evince uses the `GFileInfo` API to load, store, and modify file-specific attributes. Attributes managed by Evince have names that start with `metadata::evince::`. These attributes are loaded every time Evince opens a document.

You should add a new file-specific attribute named `metadata::evince::creator-mode`. The value of this attribute can be either `1` (Creator Mode is enabled for this file) or `0` (Creator Mode is disabled for this file). This attribute should be loaded along with all other Evince-managed attributes when a document is opened.

The `gio` command can be used to access file-specific attributes. To see the current value of the new attribute for a given file, run:

```
gio info -a metadata::evince::creator-mode <file>
```

To set a new value, run:

```
gio set <file> metadata::evince::creator-mode <value>
```

You can assume that these commands (or any other command that accesses these attributes) will not be used while Evince is running. You can also assume that the user will not set the `creator-mode` attribute to any value other than `1` or `0`.

4.2 Default Settings

Evince uses the `GSettings` API to manage its default settings. When a document is opened, any file-specific attribute which does not have a value is set to its default value.

Applications are identified in `GSettings` by a unique string called a schema. Evince uses the schema `org.gnome.Evince.Default`. You should add a new setting to this schema with the key `creator-mode`. The value of this setting can be either `true` (Creator Mode is enabled by default) or `false` (Creator Mode is disabled by default). The initial default value should be `false`.

The `gsettings` command can be used to access default settings. To see the current default value of your new setting, run:

```
gsettings get org.gnome.Evince.Default creator-mode
```

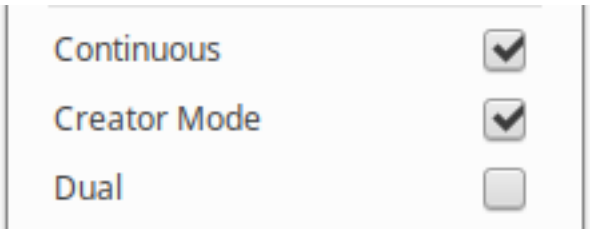
To set a new value, run:

```
gsettings set org.gnome.Evince.Default creator-mode <value>
```

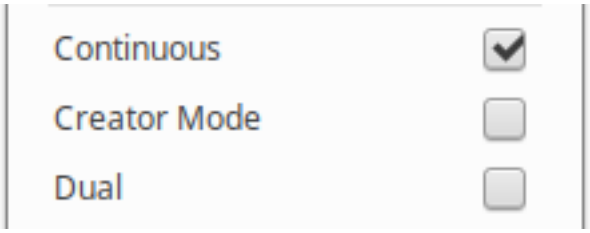
You can assume that these commands (or any other command that accesses the default settings) will not be used while Evince is running. You can also assume that the user will not set the `creator-mode` setting to any value other than `true` or `false`.

4.3 Menu Items

You should add a menu item with the label "Creator Mode". The new item should be placed right below the item labeled "Continuous". When Creator Mode is enabled for the current document, the menu item should like this:



When Creator Mode is disabled for the current document, the menu item should look like this:



When the user clicks on the new menu item, the file-specific attribute `metadata::evince::creator-mode` of the current document should change its value from `0` to `1` or vice versa.

In addition, there is a menu item labeled "Save Current Settings as Default". When this item is clicked, all file-specific attributes of the current document are saved as the default. You should add the necessary logic so that the new `creator-mode` attribute is saved as well.

4.4 Required Semantics

When Evince closes a document, it stores the current page number in the file-specific attribute `metadata::evince::page` (page numbers start at zero). When a document is opened, Evince reads the value of this attribute into an internal variable. We denote this variable by N, and the number of pages in the document by T. If the attribute is missing, N is set to zero (i.e., the first page). If $N > T - 1$ (i.e., N is past the last page of the document), N is set to $T - 1$.

After computing N, Evince does the following:

- If $N < T - 1$ (i.e., N is not the last page), the document is opened on page N.
- If $N = T - 1$ (i.e., N is the last page), the document is opened on the first page.

You should change this behavior as follows:

- If $N < T - 1$, the document should be opened on page N.
- If $N = T - 1$ and Creator Mode is enabled for the current document, it should be opened on page N (the last page).
- If $N = T - 1$ and Creator Mode is disabled for the current document, it should be opened on the first page.

4.5 Rebuilding Evince

After you modify the source code, you can build and install your modified version of Evince by running the following command:

```
make -C ~/evince/build all install
```

Then you can run Evince and test your changes:

```
evince <PDF file>
```

4.6 Testing

We recommend that you test your changes manually. Run Evince and try to use your new feature. Open some documents, enable and disable their `creator-mode` attribute, and check that Evince does what you expect.

If you insist on automated testing, there are some tools that can help. We will use Xnee and ImageMagick to grade your work. You can find other tools with similar functionality online.

5 How to Get Started

This section contains some tips and recommendations to help you get started. This is entirely optional; there are many ways to complete this assignment other than what is described here.

5.1 Opening Documents on the Last Page

First, find the code responsible for the special handling of the last page. The function `ev_window_load_job_cb()` (use `grep` to find it) is called when Evince loads a document. Read it and the functions that it calls, until you find where this special handling takes place.

When you find what you need, comment it out and rebuild Evince. Then, open a document, scroll to the last page, and close it. Open it again, and check that it opens on the last page, in contrast to the original behavior.

5.2 Document Properties

We need a way to keep track of whether Creator Mode is enabled for a given open document. A reasonable place for this is the file `libview/ev-document-model.c`. This file contains structures and functions responsible for handling several other properties of a document. Read this file and see how other properties are handled. Add a new property, which will be used to store the current state of the Creator Mode feature. Make it work like the other properties in that file. You may want to update `libview/ev-document-model.h` as well.

5.3 File-Specific Attributes and Default Settings

Read online about `GFileInfo` and `GSettings`. Search the source tree for uses of related functions. Try to understand how Evince loads these settings, where they are stored, and when they are used. Then, add a new attribute for the Creator Mode feature, as described above. Make it work like other attributes. Use it to set the new document property that you added when a document is loaded.

Now, go back to the code that you changed on Section 5.1. Modify it so that it complies with the requirements listed on Section 4.4.

Rebuild Evince and run it. Use the `gio` and `gsettings` commands to enable and disable your new feature, and check that it does what you expect.

5.4 Menu Items

Read online about `GAction`, `GMenu`, and `GtkBuilder`. Find where the window's action map is defined. Add a new action, that toggles the state of the Creator Mode feature of the current document. Use the `change_state` callback of your new action to update the corresponding document property and the file-specific attribute.

Next, find the file in which menu items are defined. If you do not know where to look, search the source tree for the top-level element in a `GtkBuilder` UI definition. Add a new menu item right after the "Continuous" item, and connect it to the window action that you just added.

Finally, figure out how the default settings are modified when the user clicks on "Save Current Settings as Default". Change the code so that your new attribute is saved there as well.

6 Final Steps

When you are ready to submit your work, you should prepare a file that contains a list of all the changes that you made. This section explains how to create this file and how to verify that it is formatted properly.

6.1 Create a Patch

If you used a version control system, use it to create a patch. Note that you need to diff against the original version, not against your last commit.

If not, run the following commands:

```
cd ~/evince
diff -Naur evince-3.36.0-{orig,new} > patch
```

The last command creates a file named `~/evince/patch`. Open this file in a text editor and review its contents. Make sure you did not forget to add anything or to remove temporary changes that you introduced for debugging.

6.2 Verify Your Patch

Create a temporary directory:

```
mkdir ~/evince/test
cd ~/evince/test
```

Extract the original source code:

```
tar xf ../evince-3.36.0.tar.xz
```

Apply your patch:

```
cd evince-3.36.0
patch -Np1 -i ../../patch
```

The last command should give no errors and its exit status should be 0. Now, build the package:

```
./configure --prefix="$HOME/evince/test/root" \
            --disable-dbus \
            --disable-nautilus \
            --without-gspell
make
```

If `make` completes successfully, then your patch is formatted properly and the package compiles with your changes. You can now remove the temporary directory:

```
cd ../..
rm -rf test
```

6.3 Write a Short Summary

You should prepare a short summary about your work. The summary should contain a list of files that you changed and a short explanation on what you changed and why. Its length should be at most 1 page with a font size of 10pt or 12pt. You should submit it in either PDF or DOC format.

6.4 Submission

You should submit a single ZIP file that contains exactly two files (no directories):

- Your patch. The name of this file must be `patch`.
- Your summary. The file name can be anything except `patch`.

Good luck! 🙂